Improvements to a User Interface for a Multi-Dimensional Data Store

[0001] The invention is practiced in the domain of databases, multi-dimensional data stores, or data warehouses, and in particular in the application of those databases, multi-dimensional data stores, or data warehouses to business planning and forecasting processes in client/server or equivalent environments.

Background

[0002] Since their invention and widespread availability, spreadsheets have provided business managers with a powerful tool to use during forecasting and planning.

[0003] Data warehouses and other repositories of large amounts of data in the form of multi-dimensional data stores are more and more a feature of business and electronic commerce, especially in enterprise-wide situations. The potential for improving business processes is large, and many tools are becoming available to assist the managers in their quest for improving forecasts. Many such tools extract data from the database or data warehouse and permit the manager to manipulate that data to perform 'what-if' operations which in turn allow the manager to explore a large number of scenarios.

[0004] An information platform carries out the following major functions, among others:

- Collects and integrates data, observations and intelligence in a data warehouse;
- Provides controls for multiple methods of information navigation and analysis;
- Allows details to be digested in the context of other data, regardless of its type.

[0005] An information platform may be implemented as a client/server system having the following four major functions.

[0006] Data gathering: The entire information platform relies upon reliable, predictable access to data, regardless of data source. The data retrieval section of the platform provides a sophisticated source of internal business information. The information platform provides users with multiple ways for collecting vast amounts and varied types of information in the shortest time possible.

[0007] Data Storage: The second section of the application platform handles the storage of the information once it has been gathered from a source. The information store uses a database (which may be object-oriented or relational) and is effectively a data warehouse. The data in the repository or warehouse are normally static, but in some instances, dynamic or volatile data may be present. This planning database which we term a planning data repository (PDR) includes the ability to change and manipulate the data, compared to the relative stability of the data in a traditional data warehouse. Both exhibit the same flexible access behaviours, i.e. users can navigate the data and find information using a variety of navigation mechanisms as described in the next section.

[0008] Information Browsing, Query, Analysis, and Report Creation: A user can peruse and analyze information contained in the PDR. The user can also make changes to some of the forecast data, and a calculation engine ensures that the changes are consistent with other data. Reports can be generated on various data.

[0009] Desktop Integration: Typically the users access the PDR from client computers in the client/server environment. These client computers might include special client applications, or they might take advantage of the latest web browser technologies to provide ubiquitous and universal access.

[0010] Electronic spreadsheets are well-known in the art and implementations having the ability to deal with thousands or even millions of cells are not uncommon. Spreadsheet programs allow users to perform "what-if" scenarios. After a set of computational relationships has been entered into a worksheet, the spread of information can be recalculated using different sets of assumptions, with the results of each recalculation appearing almost instantaneously.

[0011] It is simple and understood in the art to create a conventional spreadsheet program written in a browser scripting language, of which JavaScript is a well-known example, which can be interpreted by many standard browsers, Netscape and Microsoft Internet Explorer being typical. Such scripting languages are used to build upon and extend the capabilities of Hyper-Text Markup Language (HTML).

[0012] Further, the use of spreadsheets to interact with data from a data warehouse in a limited way is well-known and understood in the art. In such implementations, the data selected by the user is downloaded from the data warehouse (in a quasi batch-mode operation) and 'loaded' into the spreadsheet where the various formulae and functions are applied, and the results displayed for the user.

[0013] It is also known for some spreadsheet and spreadsheet-like programs to communicate with external programs and data. Such programs can, for example, export "live" data from a worksheet to a document created in a word processing program. If the data in the worksheet changes, it changes in the document as well. Similarly, a worksheet may import "live" data obtained from an external database through inter-process communication. If the data in the database changes, the change will appear in the worksheet. These links between the spreadsheet and external programs are one-way: data is either sent to the external program or received from it, but not both. In all cases, the calculations are limited to those capable of completion using only the data copied and stored locally by the spreadsheet program, without continuing reference to an underlying data repository.

[0014] Other products based on spreadsheets with separate computation capabilities have been described (e.g. US Patent 5,893,123 "System and method of integrating a spreadsheet and external program having output data calculated automatically in response to input data from the spreadsheet"), where both the nature of the underlying data and computation capability are considerably different from this invention. In the case of the '123 patent, the underlying data and the computation capability are contained in an electronic circuit simulator.

[0015] There exist products which are very similar in appearance to typical spreadsheet programs, but again they lack full integration with any underlying centralized data store. One such product, BrainMatter[1] by "AlphaBlox" is a spreadsheet application written entirely in JavaScript[2] and Hyper-Text Markup Language (D-HTML).

[0016] Gedafe (Generic Database Front-End) from Departement für Elektrotechnik, ETZ Zürich (http://www.isg.ee.ethz.ch/tools/gedafe/index.en.html) is a web-based database front-end that is database-application independent. The application code does not contain any information about what tables are present in the database or how the data is organized. This product relies on a full-featured SQL database server which permits definition not only of the format of the various tables and fields, but also of how tables are related to each other. These features allow the implementation of data integrity constraints inside the database itself so that the database server itself guarantees the integrity of the database, independently from the software used to access the database. Although such a front-end might read all the integrity constraints directly off the database and enforce them itself in order to provide faster response to the user, at the end of the day the database server will only accept data which follow the rules defined by the database programmer. Data integrity is enforced for all possible interactions with the database short of manipulation of the database structure and rules themselves, but it imposes a large load on the database server during insert and update operations. However, Gedafe is limited to maintaining database constraints that can be defined in SQL, such as foreign keys and min/max values. Further, specific application software is required at the client.

---

[1] TM of AlphaBlox

[2] Registered trademark of Sun Microsystems

[0017] Despite the obvious benefits of programs such as BrainMatter and Gedafe, they have limitations in their ability to deal with large data warehouses or Planning Data Repositories, and do not take full advantage of the possibilities provided by the ability of multiple users accessing the data, more or less simultaneously.

[0018] What is needed is a means of overcoming the limitations of existing client applications. Such improvements would allow better use of the underlying data and other information stored in a Planning Data Repository or similar facility.

Summary of the invention

[0019] Existing client applications are limited in terms of their complexity, ability to permit geographically dispersed users interact with the same database while maintaining the overall integrity of the data. Specifically, programs such as BrainMatter are limited in their ability to ensure that the data changes made by a user at the client are fully 'compatible' and consistent with the other data within the underlying database. The invention seeks to overcome these and other limitations. In doing so, it proved highly desirable and efficacious to create a two-way linkage between the client computer and the multi-dimensional database with its associated processor(s), so that the associated processor(s) can receive input data from one or more cells displayed by the spreadsheet-like program, manipulate the data within the context of the database so that it is consistent, and produce output data, based upon the input data, to be displayed in another one or more cells of the spreadsheet-like program.

[0020] In an environment consisting of a client personal computer running various applications, a communications path or network (typically an intranet or the Internet/World Wide Web), a server computer (or computer complex), also running applications, obtaining data from a planning data repository or data warehouse, itself a computer complex with large storage capacity, what is needed is the ability to allow the data displayed by the spreadsheet-like program to be updated once it has been checked for consistency in the planning data repository.

[0021] The invention ensures consistency in the underlying data in a conceptually simple manner.

[0022] The invention also has the advantage of not depending on the computing platform used by the client. Rather than compete with major desktop analysis and reporting tools, the invention makes use of generic workstations equipped with industry-standard browser software which is capable of running client-side scripts. In other words, no special hardware or software is required at the client.

[0023] One of the underlying concepts is to make use of the powerful functions or operations provided by a suitable calculation engine (of which the co-pending application - our ref. 08886651 - is an example), in collaboration with a planning data repository. The size and relative complexity of systems using data warehouse techniques, which are required for a planning data repository, means that earlier approaches involving copying data into the client, changing part of that data, computing the related updates, and confirming their consistency across the database, are not feasible.

[0024] The invention consists of a spreadsheet-like program - the client application - written in a browser scripting language, of which JavaScript is but one well-known example, which can be interpreted by many standard browsers, including Netscape Navigator[3] version 2.0 and later, Microsoft Internet Explorer[4] version 3.0 and later, and Opera version 3.0 and later. This client application interacts, possibly remotely over a network like the Internet, with a planning data repository and its associated computing resources.

[0025] While the client application has all the appearance of a conventional spreadsheet, the invention takes a somewhat different approach than that used in the

---

[3] Registered trademark of Netscape Communications Corporation.

[4] Registered trademark of Microsoft Corporation

normal storage and calculation functions of a spreadsheet. In the invention, all of the data used in computing the values shown in the spreadsheet cells are derived directly from a planning data repository (PDR) or data warehouse. Any changes made by the user to the displayed data are sent to the PDR and, when validated, incorporated into the overall data following a two-step process of 'calculate' and 'save'.

[0026] Thus much of the underlying functionality of the invention resides in the PDR and its associated processor(s). At the user's desktop machine (the client computer), the user views a window which contains information temporarily stored locally within the client computer for use by a script application. Not all of the information stored for the script application is necessarily visible on the display at any one time, and scroll bars are provided to permit the user to view information stored but not currently visible.

[0027] In some cases the data, which might otherwise be changed by the user or during processing, is locked to prevent such changes.

[0028] In one embodiment of the invention, the data shown on the client computer screen is a representation of a planning cube (or data cube) that is a subset of the Planning Data Repository data.

[0029] In a further embodiment of the invention, as with conventional spreadsheets, the user is able to change the various relationships amongst the data – in other words, the formulae relating the data may be altered. These alterations are sent to the PDR for computation, since the ability of a client work-station to handle the data and computations is limited.

[0030] In a still further embodiment of the invention, following the completion of calculation, only changed data are resent by the PDR or database server to the client, thereby saving bandwidth and/or processing capability.

[0031] The invention provides a means of interacting with a subset of a large amount of related information that assists in providing a user with the overall understanding necessary to execute rapid and knowledgeable decision-making,

especially in business planning and forecasting. The information is accessed through a combination of desktop and server technologies that raise the decision-making abilities of business professionals to the highest possible level.

[0032] In contrast to the present invention, the previously mentioned US Patent 5893123, "System and method of integrating a spreadsheet and external program having output data calculated automatically in response to input data from the spreadsheet", describes a spreadsheet program which is dependent on a database to provide the data. However, part or all of the processing is carried out in the spreadsheet application, rather than at the database. Also the nature of the data being processed is very different, with the present invention dealing with enterprise data and the prior art patent being related to circuit design and similar fields in which the data and their relationships are significantly less complex and extensive. In addition, the prior art describes the integration of data with an existing spreadsheet through the addition of processing modules. The present invention is a separate application, implemented as a client-side script and running in a web browser using industry standard facilities.

Figures

[0033] The preferred embodiment will now be described with reference to the following figures:

[0034] Figure 1 depicts a planning data repository connected over a 'network' such as the internet to a client.

[0035] Figure 2 is a representation of the way the client works, windowing over a set of data values from the planning data repository.

[0036] Figure 3 is a message flow diagram showing the steps involved in changing data at the client, performing calculations, and propagating changes to both the end user and the multi-dimensional data store.

Detailed description.

[0037] Referring first to Figure 1, wherein is shown a client computer **100** using a web browser **102,** connected over a network **110** to a Server **105** containing a Planning Data Repository (PDR) **130** which has associated with it a Calculation Engine **140** and an application process or Server **120**. When the client computer has established a logical connection to the PDR and associated software resources, information or data stored within the PDR is sent to the client computer, together with metadata describing that data.

[0038] Metadata is information (or data) about data. Simple examples of metadata collected for a data warehouse may pertain to:

- the data structure or schema
- data warehouse table attributes and structures.
- mapping from the operational database to the data warehouse
- the meaning of, or an interpretation of, the data for a business objective.

[0039] Access metadata is the dynamic link between the data warehouse and end-user applications. It generally contains the enterprise measures supported by the data warehouse and a dictionary of standard terms including user-defined custom names and aliases. Access metadata also includes the location and description of data warehouse servers, databases, tables, detailed data, and summaries along with descriptions of original data sources and transformations. Access metadata provides rules for drill up, drill down and views across enterprise dimensions and subject hierarchies like products, markets, and customers. Access metadata also allows rules for user-defined custom calculations and queries to be included. In addition, access metadata contains individual, work group, and enterprise security for viewing, changing, and distributing custom calculations, summaries, or other analyses.

[0040] Not all of the metadata is used or relevant to this invention. In particular neither the metadata relating to the data warehouse structure, nor the mapping between the data warehouse and the operational database are of immediate or direct interest to the user. The metadata used in the invention only describes the cube itself

(dimensions and members). The actual layout of the underlying database(s) is abstracted away by the relevant Application Programming Interfaces.

[0041] The data, together with relevant descriptors or labels derived from the metadata, can be displayed in a suitable multi-dimensional representation of that data, typically as a number of two-dimensional tables. Since the number of data points of interest to the user, despite being a subset of the data within the PDR, may still exceed the physical ability of the display mechanism to be shown clearly, a 'windowing' mechanism may be provided as illustrated in Figure 2, which shows an array of data **200** downloaded from the Planning Data Repository, and a smaller portion **210** of it as being rendered on the display of the users terminal, in this case using HTML, but which generally could be any suitable windowing software. The displayed window might include scroll-bars and other features typically used in such interfaces. For example, the visible window may be implemented as an HTML table within the browser, with areas along the lower and right sides denoted as scroll bars, with an appropriate script (or program) running within the browser which causes the HTML table to be re-rendered as necessary as the user scrolls over the array.

[0042] Referring now to Figure 3, a message flow of the interaction between various elements of the system, namely the Browser **300**, the Server **301**, the Calculation Engine (CE) **302** and the Planning Data Repository (PDR) **303**. Initially, the browser **300** requests certain data from the PDR. A message **305** to this effect is sent by the client **300** to the server **301**, which in turn sends a request for data **306** in the format appropriate to the planning data repository **303**. The appropriate data and metadata **307** are then sent from the PDR **303** to the server **301** and thence to the browser **300** in a further message **308** that uses a format appropriate to the browser **308**. The user, having examined the data from the PDR, is able to change some (potentially any) of the values presented on the display. When the user has decided that the changes reflect the situation they wish to examine, the Browser sends the resultant changed values with a Request Calculation **310** to the server application software - the 'Calculate' step. The server in turn formats a message **311** to the Calculation Engine **302**. A series of iterative steps **312, 313, 314, 315** involving the

PDR **303** and the Calculation Engine **302** are performed to ensure that the changes requested are self-consistent and also consistent with the other data stored in the PDR.

[0043] The Calculation Engine, described in more detail in a co-pending patent (Our ref 08-886651), is able to deal with complex planning calculations based on data warehouse or Planning Data Repository (PDR) data where some aggregated data or forecast data might be changed without directly manipulating the underlying data, and where there may be several relationships linking the data.

[0044] In considering the various formulae and functions describing these relationships, the Calculation Engine is also able to deal with complex relationships along more than one axis or dimension. A number of iterations are typically used involving both back-solving and 'forward-solving'. These relationships may be arbitrarily complex.

[0045] The advantage of the Calculation Engine described here lies in the ability to identify, before a step of back-solving and/or forward-solving, the subset of cells that needs to be recalculated. This is done using parent/child tables which simply identify and record the fact that the value in a particular cell depends on a value in one or more other cells. Once such parent/child tables exist, it is much simpler and faster to scan these tables looking for potential dependencies than to look at the actual formulae or functions relating the cells. The result is that there is the potential for huge savings in computing resources required to reach a solution in those situations where the cubes are very large, since in general, the number of cells actually affected by a given set of relationships is much smaller than the number of cells in the cube. In practical terms, the expected savings are yielded, although in some extreme and rare cases where a change needs to be propagated throughout the entire cube, and the savings may not be as large.

[0046] In general, for large complex cubes, the step of creating the parent/child tables is carried out in advance of the actual calculation by parsing all the relationships (formulae and functions) and summarising the dependencies between cells in parent/child tables.

[0047] For each rule (equation/function) or relationship the Calculation Engine completes an 'inverse back-solve' for each cell of data affected by the rule in question.

[0048] Once the calculations have been completed, and a consistent result **316** achieved by the calculation engine **302**, the new data are sent **317** to the browser **300** from the server **301**, accompanied by the relevant metadata. Note that no changes are made to the data within the PDR.

[0049] After the user has reviewed the changes, and possibly made other changes, including in some cases the removal of previous changes using an 'undo' function, the user can submit the new set of data to the PDR for a further consistency check, repeating the sequence starting with message **310** and ending with message **317**.

[0050] Finally, where the data have been reviewed and are consistent, and the user is satisfied, the user can request that the changes be incorporated in the PDR, the 'Save' step. This is achieved by the Browser **300** sending a 'request save' message **320** to the server **301**, which in turn issues a 'save' message **321** to the Calculation Engine which in its turn saves the updated data **322** in the Planning Data Repository **303**. On completion of the updating of the data the Calculation Engine **302** sends an 'acknowledge' message **323** to the Server which sends an equivalent 'acknowledge' message **324** to the browser **300**.

[0051] In cases where the requested changes are inconsistent, (or not permitted), no related changes are made and the inconsistent data are flagged to the user (for example by colour-coded changes in the displayed data, e.g., red for inconsistent data, orange for out-of-range data.)

[0052] The above description of a preferred embodiment of the invention is presented for the purposes of illustration and description. It is not intended to limit the invention to the precise form disclosed or to be exhaustive. Those skilled in the art will recognize that many modifications and variations are possible in light of the

above teaching. It is intended that the scope of the invention be limited not by this detailed description, but rather by the claims appended hereto.